22. tétel - HTTP

# Bevezetés

* tcp feletti protokoll
  + adatveszteni nem akarunk
* HyperText Transfer Protocol
* alapértelmezésben egyetlens kérés / válasz TCP-b
* stateless
  + nem tart fenn állapotot a szerver az egyes kliensekről
  + nem erre készült, de ez terjedt el
  + egyszerű, de nem tud sokat -> cookies
  + sütik nélkül minden weboldal lekérés egymástól független lenne

# Cookie

* a cookiek segítségével a szerver bekukucskál a gépünkbe
* információcsomag
* legtöbbször azonosítás, bevásárló kosár megtartására, nyomonkövetésre használják
* gyakran reklámokkal jön -> célzott reklámokat akarnak
* ha rossz kezekbe kerül szopás : XSS

# HTTP kérések és válaszok

* kérés és válasz szerkezete megegyezik
* a kezdő sor megmondja milyen kréés vagy milyen válasz



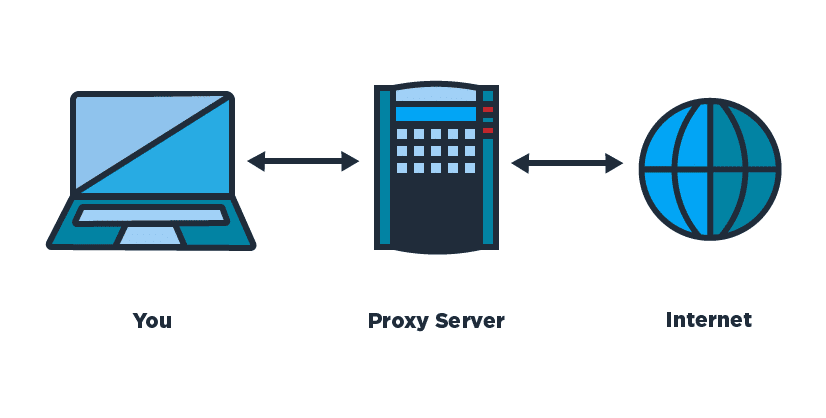
* a leggyakorib http kérés : GET
* HEAD
  + Ugyanazt adja vissza, mint a GET, csak magát az üzenettestet hagyja ki a válaszból.
* GET
  + A megadott erőforrás letöltését kezdeményezi. Ez messze a leggyakrabban használt metódus.
* POST
  + Feldolgozandó adatot küld fel a szerverre. Például HTML űrlap tartalmát. Az adatot az üzenettest tartalmazza.
* PUT
  + Feltölti a megadott erőforrást.
* DELETE
  + Törli a megadott erőforrást.
* TRACE
  + Visszaküldi a kapott kérést. Ez akkor hasznos, ha a kliens oldal arra kíváncsi, hogy a köztes gépek változtatnak-e illetve mit változtatnak a kérésen.
* OPTIONS
  + Visszaadja a szerver által támogatott HTTP metódusok listáját.
* CONNECT
  + Átalakítja a kérést transzparens TCP/IP tunnellé. Ezt a metódust jellemzően SSL kommunikáció megvalósításához használják.

# HTTP/1.1 sajátságok

* Virtuális szerverek
  + egy IP-n több szervert lehet hostolni
  + Ezért létezik a Host fejléc elem
* Chunked Transfer Encoding
  + a fejrész elemek között jelzi a szerver a kontent hoszát
  + darabokban küldi az információt
* Prezisztens kapcsolatok
  + A http/1.1-ben egy TCP kapcsolaton több kérés/válasz pár is végbemehet

# Proxy

* szerverként és kliensként is működik
* köztes elem
* pl gyorsabb weboldal betöltés - cachel információt
* anonimitás



# SSL/TLS

* Netscape találmány
* Secure Socket LAyer
* nyilvános kulcsú titkosítás
* védelem a böngészés felett
* lehallgatás megakadályozása

# TLS handshake protokoll

* Kriptográfiai algoritmusokban és paraméterek egyeztetése
* Véletlen számok szerint 'master secret-et' generálnak
* A TLS record réteg számára paramétereket adnak át
  + Rendszeresen cserélik a titkosításra használt kulcsot
* A kliens 'Client hello' üzenettel köszön
  + Közli, hogy milyen protokollt fog használni (pl. SSL vagy TLS)
  + Közli, hogy milyen algoritmusokat támogat
* A szerver 'Server hello' üzenettel válaszol
  + A ,,legnagyobb közös osztó'' lesz a használt protokoll…
* sok verzió volt, botrány man inthe middle